INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 6, June 2026 | ISSN: 2456-4184 | IJNRD.ORG

A Comparative Study of Error Detection & Error Correction Methods in Computer Networks

[1]Shailesh Mohan Shukla, [2]Bhumika Nayal, [3]Manoj Kumar

[1]Student, [2]Student, [3]Assistant Professor
[1,2&3]Computer Science & Engineering Department,
[1]JB Institute of Technology, Dehradun, India, [2]Graphic Era Hill University, Bhimtal, India, [3]JB Institute of Technology, Dehradun, India

Abstract: Reliable data transmission is a fundamental requirement in computer networks. During data communication, errors may occur due to noise, interference, or hardware failures. Error detection methods are therefore essential to ensure data integrity. This research paper presents a comparative study of widely used error detection techniques, including parity checking, cyclic redundancy check (CRC), checksum, and two-dimensional parity. The study evaluates these methods based on efficiency, error detection capability, computational complexity, and practical applicability**.**

Keywords: CRC, Checksum, FEC, Error Detection, Parity Check, Error Correction.

1. INTRODUCTION

In modern computer networks, accurate data transmission is critical for applications ranging from web browsing to real-time communication systems. However, during transmission, data may be corrupted due to various factors such as electromagnetic interference, signal attenuation, and hardware faults. These errors can lead to incorrect information being received, which may cause serious issues in critical systems.

Error detection techniques are designed to identify whether transmitted data has been altered during transmission. These techniques do not necessarily correct errors but help in identifying them so that retransmission can occur.

2. LITERATURE SURVEY

Early work by Claude Shannon (1948) laid the foundation for error control coding through his theory of communication, which established the concept of channel capacity and demonstrated that reliable communication is possible even in the presence of noise. This work motivated the development of both error detection and correction techniques.

One of the earliest and simplest error detection methods, parity checking, has been discussed in detail by Stallings (2017) and Tanenbaum and Wetherall (2011). These studies highlight its low computational complexity but also emphasize its limitations in detecting multiple-bit errors. To overcome these limitations, more advanced detection techniques such as Cyclic Redundancy Check (CRC) were introduced.

The concept of CRC was formalized by Peterson and Brown (1961), who demonstrated that polynomial-based error detection methods are highly effective in identifying burst errors. CRC has since become a standard technique used in network protocols such as Ethernet and data link layer communication. Further analysis by Forouzan (2013) explains the mathematical foundation and practical implementation of CRC in modern networks.

Checksum methods have also been widely studied, particularly in the context of Internet protocols. According to Kurose and Ross (2017), checksum provides a balance between simplicity and performance, making it suitable for protocols like TCP and UDP. However, researchers note that checksum is less reliable compared to CRC in detecting complex error patterns.

In addition to error detection, significant research has been conducted on error correction techniques. Hamming (1950) introduced Hamming codes, which are capable of both detecting and correcting single-bit errors. This marked a major advancement in error control, as it eliminated the need for retransmission in certain cases.

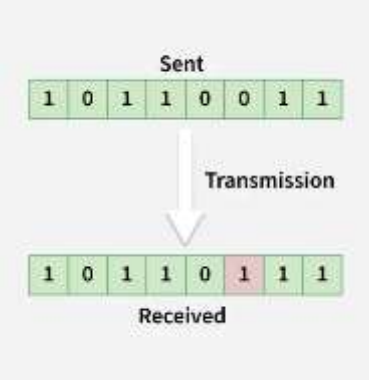
Further developments in error correction were made with the introduction of Reed-Solomon codes, which are particularly effective in correcting burst errors. Studies by Lin and Costello (2004) and Blahut (1983) provide detailed insights into the mathematical models and applications of these codes in digital communication and storage systems.

Forward Error Correction (FEC) techniques have gained importance in modern communication systems, especially in wireless and satellite networks where retransmission is costly. According to Moon (2005), FEC improves system reliability by adding redundancy, allowing the receiver to correct errors independently.

TYPES OF ERRORS

1. SINGLE-BIT ERROR

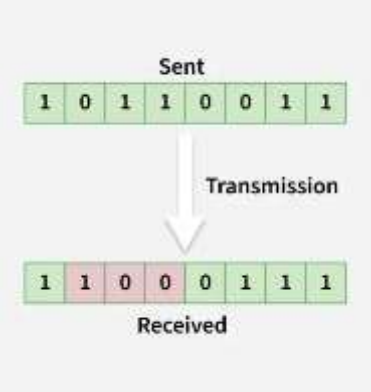
A single-bit error occurs when only one bit of the transmitted data unit is altered during transmission, resulting in corrupted data.



Single-Bit Error

2. BURST ERROR

A burst error occurs when two or more consecutive bits in a data unit are corrupted during transmission.



Burst Error

ERROR DETECTION METHODS

To detect errors, a common technique is to introduce redundancy bits that provide additional information. Various techniques for error detection include:

- Simple Parity Check
- Two-Dimensional Parity Check
- Checksum
- Cyclic Redundancy Check (CRC)

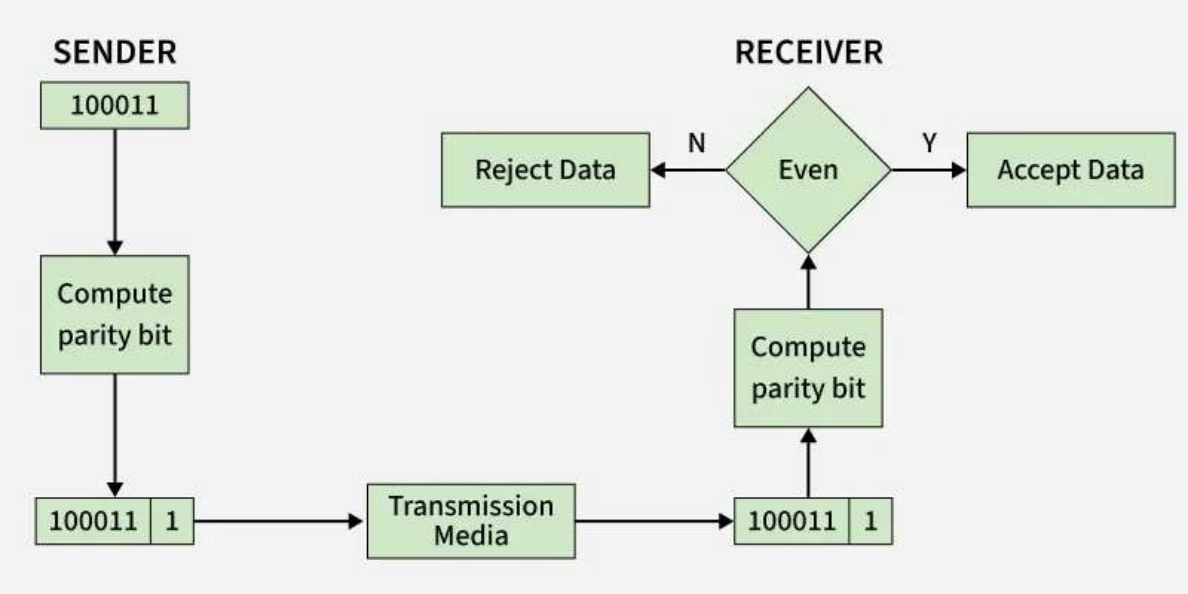
1. SIMPLE PARITY CHECK

Simple bit parity is a basic error detection technique in which an extra bit, called a parity bit, is added to a data unit before transmission. This parity bit helps the receiver determine whether the transmitted data has been corrupted.

- Detects all single-bit errors in transmitted data.
- Detects any odd number of bit errors, since odd bit changes alter the parity.
- Easy to implement in both hardware and software.
- Fails to detect errors when an even number of bits are corrupted.
- Not suitable for noisy communication channels with frequent errors.

Parity bits are used to maintain a specific parity condition in the data. There are two types of parity:

- Even Parity: The parity bit is set so that the total number of 1s in the data (including the parity bit) is even.
- Odd Parity: The parity bit is set so that the total number of 1s in the data (including the parity bit) is odd.



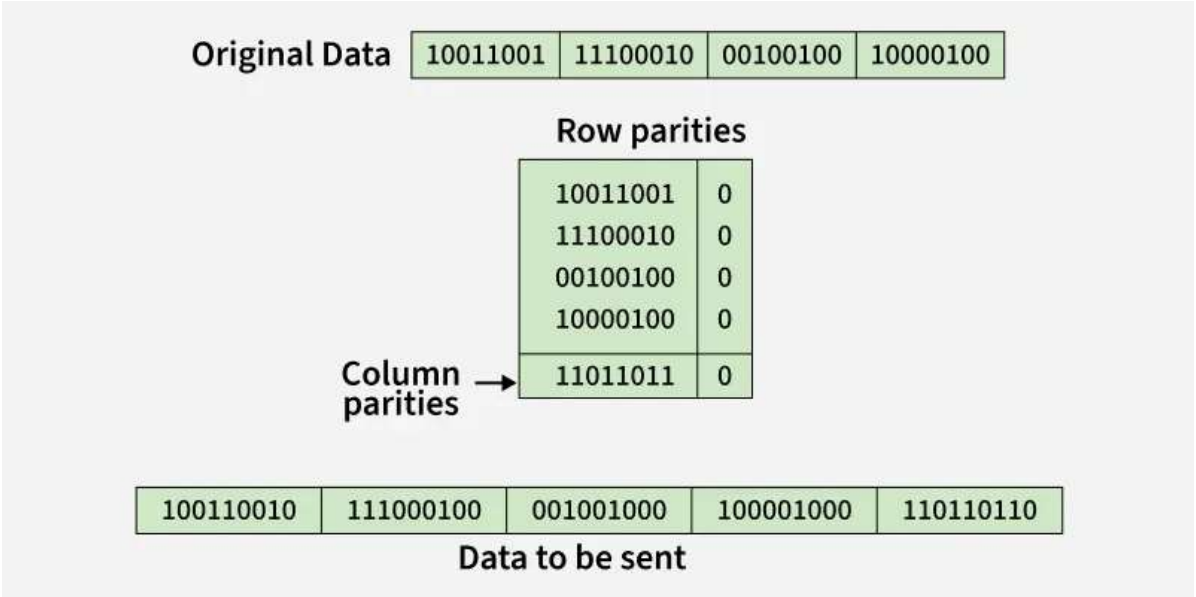
Simple Parity Check

This scheme makes the total number of 1's even, that is why it is called even parity checking.

2. TWO-DIMENSIONAL PARITY CHECK

In two-dimensional parity check, parity bits are calculated for each row, similar to a simple parity check. In addition, parity bits are also computed for each column. These row and column parity bits are transmitted along with the data. At the receiver, parity bits are recalculated and compared with the received parity bits to detect errors.

- Can detect and correct all single-bit errors by identifying the exact row and column where the error occurred.
- Can detect many multiple-bit errors occurring at different positions in the data matrix.
- Provides better error detection capability than simple parity check.
- Certain patterns of multiple-bit errors may remain undetected.
- If parity bits themselves are corrupted, error detection may fail.



2-D Parity Check

3. CHECKSUM

Checksum is an error detection technique used to detect errors in transmitted data. In this method, the sender divides the data into fixed-size segments and computes a checksum using 1's complement arithmetic. The computed checksum is transmitted along with the data. At the receiver, the same computation is performed to verify data integrity.

- Can detect many types of errors, including single-bit and multiple-bit errors.

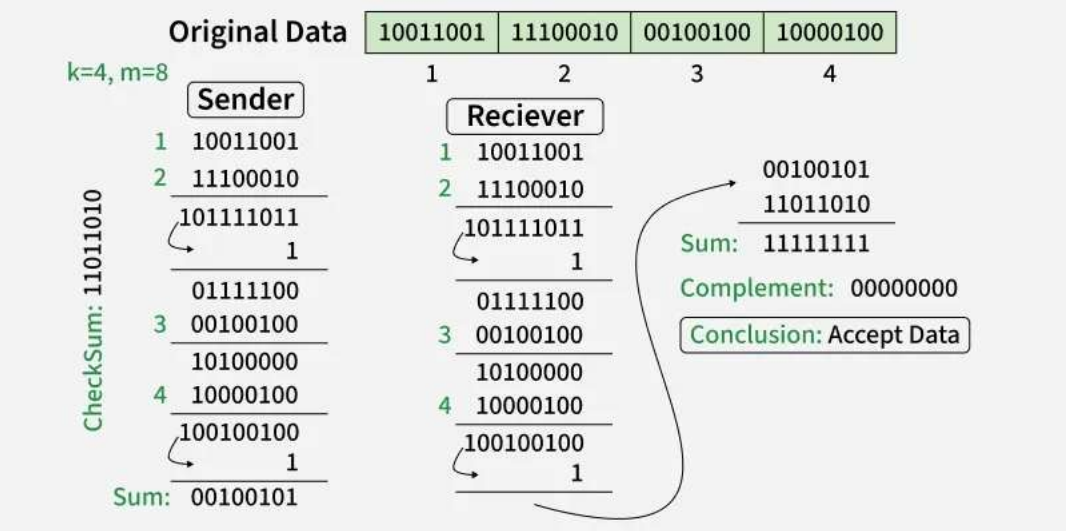
- Provides better error detection than simple parity and two-dimensional parity checks.
- Widely used in networking protocols such as IP, TCP, and UDP.
- Fast to compute and easy to implement in software.
- Supports only error detection and does not provide error correction.
- Some error patterns may go undetected, especially when errors cancel each other during addition.
- Less reliable compared to advanced techniques like Cyclic Redundancy Check (CRC).

Checksum - Operation at Sender's Side

- The data is divided into k segments, each of m bits.
- All segments are added using 1's complement arithmetic.
- The 1's complement of the final sum is calculated to generate the checksum.
- The checksum is appended to the data and transmitted to the receiver.

Checksum - Operation at Receiver's Side

- All received segments, including the checksum, are added using 1's complement arithmetic.
- If the final result is all 1s, the data is considered error-free.
- Otherwise, the data is discarded, indicating that an error has occurred.



Checksum at Receiver 's and Sender's Side

4. CYCLIC REDUNDANCY CHECK (CRC)

- CRC is based on binary division, unlike checksum which uses addition; redundant CRC bits are appended so the transmitted data becomes exactly divisible by a predefined generator polynomial.

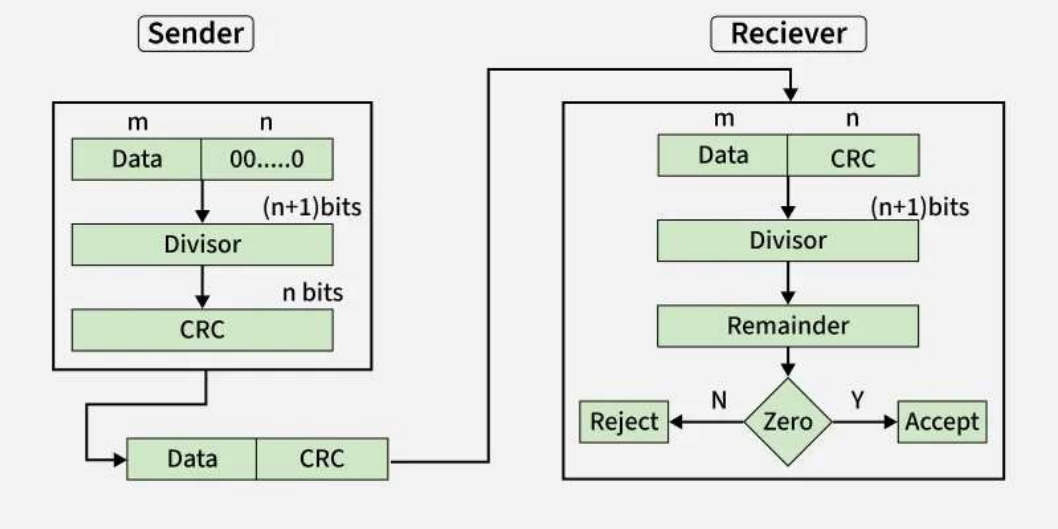
- At the receiver, the data is divided using the same polynomial; a zero remainder means the data is accepted, while a non-zero remainder indicates transmission errors.

- Highly effective in detecting single-bit, multiple-bit, and burst errors, making it more reliable than parity checks and checksum methods.

- Widely used in communication protocols such as Ethernet, HDLC, and USB due to its strong error detection capability.

- Provides only error detection, not error correction; its effectiveness depends on the chosen generator polynomial.

- More computationally complex than simple parity or checksum methods and introduces additional processing overhead.



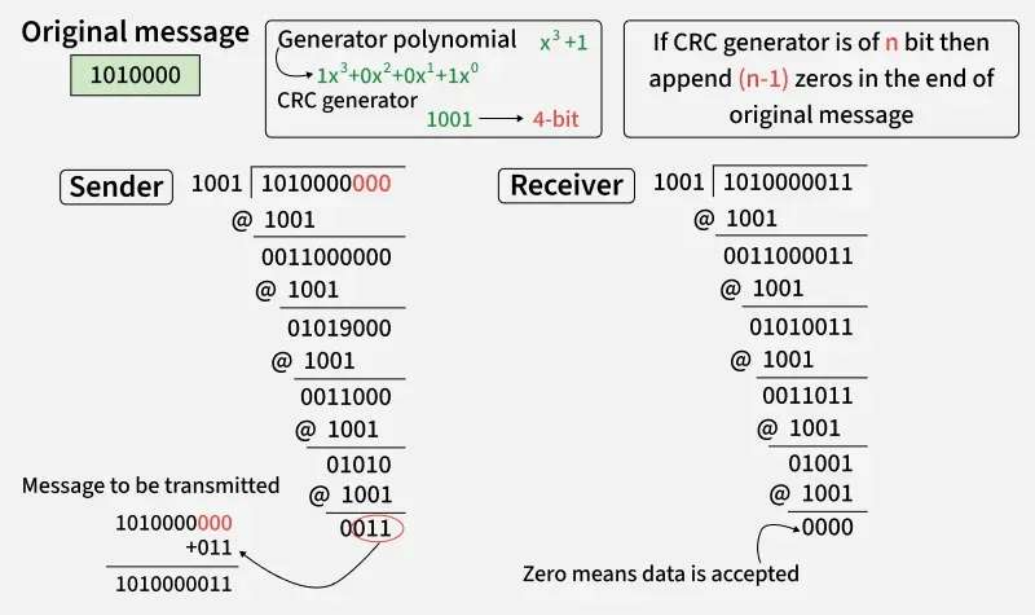
We have given data word of length n and divisor of length k.

Step 1: Append (k-1) zeroes to the original message
Step 2: Perform modulo 2 division
Step 3: Remainder of division = CRC
Step 4: Code word = Data with append k-1 zero's + CRC

Note:

- CRC must be k-1 bits
- Length of Code word = n+k-1 bits

Example: Let's data to be send is 1010000 and divisor in the form of polynomial is x3+1. CRC method discussed below.



5. COMPARATIVE ANALYSIS

Method	Error Detection Capability	Complexity	Overhead	Suitable Use Case
Parity Checking	Low	Very Low	Minimal	Simple systems
Two-Dimensional Parity	Medium	Low	Moderate	Moderate systems
Checksum	Medium	Medium	Moderate	Network protocols

Method	Error Detection Capability	Complexity	Overhead	Suitable Use Case
CRC	High	High	Moderate	High-reliability systems

The comparison shows that CRC outperforms other methods in detecting errors, especially burst errors. However, it comes at the cost of increased complexity.

6. RESULTS AND DISCUSSION

The analysis reveals that no single error detection method is universally optimal. The choice of method depends on system requirements.

- Low-resource systems benefit from parity checking due to its simplicity.
- Moderate systems may use checksum or two-dimensional parity.
- High-performance networks require CRC for reliable communication.

CRC is particularly effective in detecting burst errors, which are common in real-world communication channels. Its widespread adoption in Ethernet and storage systems highlights its reliability.

However, simpler methods still have relevance in systems where computational resources are limited. For example, embedded systems may prefer parity checking due to its low overhead.

7.ERROR CORRECTION IN COMPUTER NETWORKS

Error correction is the process of detecting and fixing errors that occur during data transmission so that the receiver can correctly understand the received data. Errors may happen due to noise, interference, or other transmission issues. Error correction techniques help maintain data integrity and ensure reliable communication between network devices. These techniques allow the system to recover correct information even when some transmitted bits are corrupted.

1. Forward Error Correction (FEC):

- Forward Error Correction is an error control technique used during data transmission.
- The sender adds extra (redundant) bits to the original data before sending it.
- These extra bits help the receiver detect and correct errors on its own.
- The receiver does not need to request retransmission from the sender.
- This reduces delay and improves communication reliability.
- FEC is useful where retransmission is costly or not possible.
- It is commonly used in wireless networks, satellite communication, and multimedia streaming.
- Hamming Code is a simple and common example of a Forward Error Correction technique.

2. Backward Error Correction:

- Backward Error Correction is an error control technique used in data communication.
- The receiver checks the received data for errors using error detection methods.
- If an error is found, the receiver sends a retransmission request to the sender.
- The sender retransmits the corrupted data until it is received correctly.
- This method requires a feedback channel between the sender and receiver.
- It provides reliable data delivery but can increase transmission delay.
- Backward Error Correction is commonly implemented using Automatic Repeat Request (ARQ) techniques.
- Common ARQ methods include Stop-and-Wait, Go-Back-N, and Selective Repeat.

3. Types of Forward Error Correction (FEC)

1. Hamming Code

Hamming Code is a Forward Error Correction (FEC) technique used to make data transmission more reliable. It allows the receiver to detect and correct errors on its own without asking the sender to resend the data.

- Hamming Code can correct one-bit errors and detect two-bit errors, but it cannot fix two errors at the same time.

- It is suitable for systems where errors are rare and usually occur one bit at a time.2. Reed–Solomon Code
- Parity bits are placed at positions 1, 2, 4, 8, 16, …, which are powers of two.
- Each parity bit checks a specific group of data bits, helping locate the exact position of an error.
- The receiver uses parity checks (syndrome) to identify the incorrect bit and correct it if needed.
- Since the receiver can fix the error by itself, no retransmission is required, making it a true Forward Error Correction technique.

2. Reed–Solomon Code

Reed–Solomon Code is a block-based Forward Error Correction (FEC) technique widely used in digital communication and data storage systems. It works on symbols instead of individual bits, where each symbol contains multiple bits. Because of this symbol-based approach, Reed–Solomon codes are very effective at correcting burst errors, in which many consecutive bits are damaged during transmission or storage.

- Symbol-based working: Reed–Solomon codes work on data blocks made of symbols, where each symbol contains multiple bits. Extra parity symbols are added to the block for error correction.
- Multiple error correction: The number of errors that can be corrected depends on the extra symbols added. If 2t redundant symbols are used, the code can correct up to t corrupted symbols in a block.
- Stronger than bit-level codes: Since errors are corrected at the symbol level, Reed–Solomon is more powerful than simple methods that correct only single-bit errors.
- Very effective for burst errors: Burst errors usually damage several consecutive bits, often affecting full symbols. Reed–Solomon codes are designed to recover such damaged symbols accurately.
- Wide practical usage: These codes are widely used in CDs, DVDs, Blu-ray discs, QR codes, barcodes, satellite communication, and digital TV, where data accuracy is critical.
- Higher computation cost: Reed–Solomon codes need more complex encoding and decoding compared to simpler techniques like Hamming Code, but this complexity provides better error correction strength.

3. Types of Backward Error Correction

1. Stop-and-Wait ARQ

- Stop-and-Wait ARQ is a basic Backward Error Correction method for reliable data transmission.
- The sender transmits one frame at a time and waits for an acknowledgment (ACK) from the receiver.
- The receiver sends an ACK only if the frame is received correctly, ensuring reliable delivery.
- If a frame is lost, contains errors, or the ACK is not received, the sender retransmits the same frame.
- A timer is used for each frame; if it expires before receiving an ACK, the frame is resent.
- This method is simple and reliable, but inefficient for high-speed or long-delay networks due to idle waiting time.

2. Go-Back-N ARQ

- Go-Back-N ARQ is a Backward Error Correction method that allows the sender to transmit multiple frames consecutively without waiting for individual acknowledgments.
- It uses a sliding window at the sender to control how many frames can be sent before receiving ACKs.
- The receiver accepts frames only in order; out-of-order frames are discarded even if they are correct.
- The receiver sends cumulative acknowledgments indicating the highest in-order frame received successfully.
- If a frame is lost or corrupted, the sender retransmits that frame and all subsequent frames in the window.
- Go-Back-N improves throughput over Stop-and-Wait but can waste bandwidth if errors occur frequently.

4. Applications of Error Correction

1. Wireless and Mobile Communication (4G/5G, Wi-Fi):

- Error correction ensures reliable data transmission over wireless channels, which are prone to interference, fading, and noise.
- Techniques like Convolutional Codes, Turbo Codes, and LDPC codes are commonly used in 4G/5G networks and Wi-Fi to improve throughput and reduce packet loss.

2. **Satellite and Deep-Space Communication**:

- Communication over long distances, such as satellite links or deep-space probes, experiences significant signal attenuation and delays.
- Forward Error Correction methods like Reed–Solomon and Convolutional Codes allow receivers to correct errors without relying on retransmissions, which are impractical due to high latency.

3. Data Storage Devices (HDD, SSD, CDs, DVDs):

- Error correction ensures the integrity of stored data, compensating for physical defects, scratches, or degradation over time.
- Codes such as Hamming Code, Reed–Solomon, and BCH codes are embedded in storage devices to detect and correct errors automatically.

4. **QR Codes, Barcodes, and Digital Broadcasting**:

- QR codes and barcodes use error correction (typically Reed–Solomon) to recover data if part of the code is damaged or obscured.
- Digital TV, radio, and streaming services implement FEC techniques to maintain high-quality transmission even in noisy environments.

8. Conclusion

Error control is a fundamental requirement in computer networks to ensure reliable and accurate data transmission. This study examined both error detection and error correction techniques, highlighting their roles, advantages, and limitations in modern communication systems.

in detecting burst errors and its widespread adoption in real-world protocols like Ethernet. Simpler methods, while computationally efficient, offer limited reliability and are suitable only for low-complexity systems.

The comparison reveals that there is a trade-off between efficiency, complexity, and reliability. Error detection methods are generally simpler and more resource-efficient, making them suitable for systems where retransmission is feasible. In contrast, error correction methods provide higher reliability at the cost of increased complexity and bandwidth usage.

In conclusion, the choice between error detection and error correction depends on the specific requirements of the network, including bandwidth availability, error rate, and system constraints. Modern communication systems often employ a combination of both techniques to achieve optimal performance, ensuring both accuracy and efficiency in data transmission.

9. References

[1] W. Stallings, Data and Computer Communications, 10th ed. Pearson, 2017.

[2] A. S. Tanenbaum and D. J. Wetherall, Computer Networks, 5th ed. Pearson, 2011.

[3] B. A. Forouzan, Data Communications and Networking, 5th ed. McGraw-Hill, 2013.

[4] J. F. Kurose and K. W. Ross, Computer Networking: A Top-Down Approach, 7th ed. Pearson, 2017.

[5] W. W. Peterson and D. T. Brown, “Cyclic codes for error detection,” Proceedings of the IRE, vol. 49, no. 1, pp. 228–235, 1961.

[6] S. Lin and D. J. Costello, Error Control Coding, 2nd ed. Pearson, 2004.

[7] R. E. Blahut, Theory and Practice of Error Control Codes. Addison-Wesley, 1983.

[8] T. K. Moon, Error Correction Coding: Mathematical Methods and Algorithms. Wiley, 2005.

[9] M. Schwartz, Telecommunication Networks: Protocols, Modeling and Analysis. Addison-Wesley, 1987.

[10] N. Abramson, “The ALOHA system—another alternative for computer communications,” Proc. Fall Joint Computer Conf., 1970.

[11] IEEE Standard for Ethernet, “IEEE Std 802.3,” IEEE, 2018.

[12] ISO/IEC 3309, “Information technology—Telecommunications and information exchange between systems—High-level data link control (HDLC) procedures,” ISO, 1993.

[13] R. Gallager, Information Theory and Reliable Communication. Wiley, 1968.

[14] C. E. Shannon, “A mathematical theory of communication,” Bell System Technical Journal, vol. 27, 1948.

Copyright & License:
© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.